International Journal of Communication Networks and Information Security
2024, 16(S1)
ISSN: 2073-607X, 2076-0930
https://https://ijcnis.org/ **Research Article**

Strengthening IOT Security by Using Ensemble Learning and Feature Selection for Intelligent Intrusion Detection Based on Complete UNSW-NB15 And Iotid20 Datasets

Pritimayee Satapathy[1*], Prafulla Kumar Behera[2]

[1*]Department of Computer Science and Applications, Utkal University, Vani Vihar, Bhubaneswar 751004, Odisha, India.
[2]Department of Computer Science and Applications, Utkal University, Vani Vihar, Bhubaneswar 751004, Odisha, India.

Email: [2]pkbehera.cs@utkaluniversity.ac.in
*Correspondence: [1]pritimayees@gmail.com

ARTICLE INFO

Received: 19 Apr 2024
Accepted: 28 Aug 2024

ABSTRACT

Internet of Things (IoT) refers to a collection of devices with sensors or actuators, capable of sending and receiving data that are linked together over wireless networks. As the number of internet-connected gadgets is growing at a brisk pace, a huge volume of data is being passed through these devices. This calls for the development and optimization of algorithms related to network security such as Intrusion Detection Systems (IDSs) to keep data secure during transmission. As a result, IDSs are frequently used along with additional safety solutions like firewalls and access control for data security. Various Machine Learning (ML) based strategies have been used for customizing IDSs to meet the ever-increasing demands of secured networks based on a subset of IoT Intrusion detection dataset. In the present study, we implemented an ensemble ML technique applied to the full version of IoT Intrusion Dataset 2020 (IoTID20) as well as UNSW-NB15 datasets to carry out multiple experiments. ML methods including, Logistic Regression (LR), K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Random Forest (RF), and Extreme Gradient Boosting (XGBoost) have been used to build the primary classifier and the meta-classifier respectively in the proposed model. ML-based IDSs often encounter challenges such as a rise in the false positive rate, reduced detection accuracy, and prolonged model-building time when training is conducted on several imbalanced datasets. We implemented a random forest classifier's feature importance score to evaluate all of the pre-processed data and generate a shortened feature set, which simplifies the process of creating models and increases detection accuracy. The evaluation assessment shows that the recommended approach exhibits superior performance when compared to the models published in the literature, achieving 99.14% detection accuracy on UNSW-NB15 datasets with only 12 out of 47 network features and 99.94% detection accuracy on IoTID20 datasets with only 16 out of 83 network features.

Keywords: Internet of Things; machine learning; feature engineering UNSW-NB15; IoTID20; network security; cybersecurity; anomaly detection; intrusion detection system.

INTRODUCTION

Over the past few decades, there has been substantial advancement in the adoption of the Internet of Things (IoT) in various domains such as industrial sectors, effective transportation systems, automated agriculture, and healthcare, resulting in significant effects on socioeconomic development. IoT systems are comprised of a substantial quantity of interlinked sensors, actuators, and various network-enabled gadgets that facilitate the sharing of diverse data types across both public and private networks. According to forecasting, the IoT market will continue to grow, achieving an annual turnover of more than $2.4 trillion devices by 2027. IoT technologies exhibit distinct characteristics compared to conventional internet technology due to the absence of human intervention in the data-sharing process within the context of IoT (1) (2). The use of IoT devices has significantly increased according to a study by Ullah S. et al. (3). There has been an increase in the quantity of these gadgets, escalating from 15.41 billion in 2015 to surpassing 35.8 billion in 2021. This figure is projected to rise to 75.44 billion devices by 2025.
The tremendous growth of IoT devices and the widespread transfer of data across networks have significantly increased the chance of cyberattacks. Cyberattacks carried out with malicious intentions provide significant challenges to network security. The relevance of security, privacy, and availability in IoT systems is rising as the lines between the physical and digital worlds continue to converge. However, traditional firewall systems fail not only to detect modern attacks but also to analyze network packets in-depth (4). Traditional safety measures are hard to use on most IoT devices because they don't have a lot of resources. Because of this, the IoT environment needs to have a strong security system that can make sure networks are safe and secure. The organization must utilize IDS specifically developed for IoT environments in order to detect and address any security vulnerabilities before they can be exploited. An IDS doesn't usually protect the system from attacks. Instead, it sounds like a warning when it finds an intrusion in the network, either right away or before the attack gets to the devices it's meant to protect. IDSs are mostly divided into three groups based on how they interfere with network traffic: host-based or node-based, network-based or distributed, and hybrid IDSs. It was designed so that the node-centric IDS can find and identify threats that come from a single computer system. Network-based IDS have been put on network routers and switches to keep an eye on traffic and find harmful data in a distributed computer environment. Hybrid IDS can be used in both individual computers and network setups (5).

Based on the attack detection, all IDS can be divided into the following groups: signature-based IDS, anomaly-based IDS, and hybrid-based IDS (2). To efficiently detect attacks Signature-based IDSs use pre-established patterns or signatures. To detect anomalies, anomaly-based IDS first establishes a norm of permitted operation for connected devices and infrastructures. Finally, hybrid-based IDS uses both anomaly-based and signature-based IDSs (6).

Researchers have proposed a variety of approaches for identifying attacks on networks by using many ML techniques. A concise overview of academic efforts related to the analysis of the UNSW-NB15 dataset is provided in Supplemental Table 1. Supplementary Table 2 provides a concise overview of important research achievements related to the study of the IoTID20 dataset. In the field of IDS Al-Zewairi M. et al. (7) designed Deep Learning (DL) models utilizing the H2O platform. The developed models were evaluated using an Artificial Neural Network (ANN) utilizing backpropagation and Stochastic Gradient Descent (SGD) with an accuracy of 98.99% using the UNSW-NB15 dataset.

Agarwal A. et al. (8) utilized three well-known ML methods, namely Naive Bayes (NB), SVM, and KNN to develop an IDS. For evaluating their models, they used a total of 700001 cases, each containing 48 features which is a portion of the complete UNSW-NB15 dataset. Proposed IDS achieved an accuracy of 97.777%, 93.333%, and 95.55555 on SVM, KNN, and NB ML models, respectively. For creating an IDS Kasongo SM. et al. (9) suggested a filter-based Feature Selection (FS) technique, utilizing the XGBoost algorithm to improve detection accuracy and decrease the FPRs. For evaluating the UNSW-NB15 dataset's binary and multiclass classification they used a portion of the UNSW-NB15 dataset that was produced by Moustafa N.et al. (10). By conducting various experiments utilizing the ML methods such as KNN, LR, ANN, SVM and Decision Tree (DT), they achieved an accuracy of 88.13% using all features and accuracy of 85.855 using only 19 features in a binary classification scenario. By utilizing ensemble learning with lightweight ML methods namely Gaussian Naïve Bayes(GNB), LR and DT with meta-classifier SGD. Thockchom N. et al. (11) suggested an IDs framework. To extract the most significant features they used the Chi-square FS. For evaluating the models, they used a small portion of three datasets namely KDD Cup 1999, CIC-IDS2017, and UNSW_NB15. Rajagopal S. et al. (12) suggested an IDS using stacking-based ensemble learning with the meta-classifier concept. The most useful attributes are selected by using both Information Gain and hashing. Moustafa N. and Slay J. (13) recommended a hybrid approach that utilizes an Association Rule Mining algorithm in conjunction with the midpoints of feature values to improve the IDS and lower the False Alarm Rates(FAR). The experimental analysis was carried out using two datasets namely UNSW-NB15 and NSLKDD by utilizing various ML techniques and they found that LR achieved an accuracy of 83.0% on UNSW-NB15 and 82.1% on NSL-KDD. Thakkar A. and Lohiya R. (14) developed an FS by combining the statistically important features with variations of mean and median utilizing a Deep Neural Network(DNN). They used three IDS datasets namely NSL-KDD, UNSW-NB15, and CIC-IDS-2017. By using DNN Vinayakumar R. et al. (15) proposed an IDS through the utilization of DL techniques. They trained their models by employing five distinct IDS datasets.

Chew YJ. et al. (16) applied the rolling-origin resampling method for the distribution of both training and testing on three different IDS datasets namely GureKDDCup, UNSW-NB15, and CIDDS-001. Ten different ML classifiers were tested using the Weka tool and the J48 classifier outperformed with an accuracy of 96,63%. A hybrid FS was proposed by Albulayhi K. et al. (17) by integrating both the IG and Gain Ratio (GR) methods. Training and testing were done based on using four ML algorithms, specifically bagging, multilayer perception, J48, and IBk, using two datasets: IoTID20 and the NSL-KDD. By employing the principles of intersection and union from mathematical set theory, the researchers were able to identify 11 and 28 important features respectively, out of a total of 86, utilizing the IoTID20 dataset. Song, Y. et al. (18) conducted experiments on three distinct datasets, namely NSL-KDD, IoTID20, and N-BaIoT using a stacked autoencoder. The researchers made a significant observation that autoencoders exhibited exceptional performance when used to benchmark datasets for IDSs in the IoT domain with an accuracy of 99%.

Islam N. et al. (19) developed an IDS that implemented DL methods and compared the analysis with shallow ML techniques. They recommended that DL models perform well as compared to ML methods. They evaluated their models by using IDS datasets namely NSL-KDD, IoTDevNet, DS2OS, IoTID20, and IoT Botnet dataset. By implementing a Single Hidden Layer Feedforward Neural Network (SLFN), Qaddoura R. et al. (20) proposed an approach to identifying malicious activity in IoT networks. The researchers employed a data reduction approach by utilizing clustering techniques in conjunction with the SMOTE oversampling method. The model was assessed through the use of measures including G-mean, accuracy, precision, and recall.

As shown in the supplementary Tables 1 and 2 IDSs commonly employ ML techniques to effectively handle large volumes of data with high dimensions. However, further investigation is necessary to evaluate the effectiveness as well as the accuracy of various ML algorithms developed for ID. One of the main obstacles to the successful application of the ML technique in IDS is the abundance of features available in IoT datasets, some of which may be irrelevant. Albulayhi K. et al. (17) identified some limitations associated with the utilization of all feature sets. The computational workload is increased, resulting in longer durations for both training and testing processes. The accuracy of the detection rate is also hampered by the irrelevant features. FS algorithms play an important role in identifying attacks within the IoT network by using ML techniques. With the use of FS methods, several features of the model can be improved such as training time, FBR, detection rate, and use of resources. The application of FS also simplifies the creation of IDSs within the limited resources of IoT devices. To simplify the framework of the IoT system and find out the most significant features, we utilized the feature priority score of the RF classifier.

Numerous studies have demonstrated that the application of ensemble approaches improves the performance of individual ML algorithms, resulting in an overall increase in detection accuracy (12) (11). In the present research study, we employed a stacking-based ensemble approach in conjunction with feature engineering and normalization techniques to improve the efficiency of models in the context of IoT IDSs. Existing ML-based solutions on the UNSW-NB15 dataset suffer from the common issue that most models only make use of a hand-picked subset of the dataset provided by the data's originators. The initial creators of the datasets divided a small fraction into a training set and a test set while creating the datasets (10) (4). Despite this, very few evaluations have been performed on the full version of datasets. Furthermore, the significance of hyperparameter adjusting has often been overlooked in the majority of studies undertaken to enhance IDS. Prusa, J et al. (21) also conducted investigations to see if adding more data would improve the model's accuracy. Their studies showed that expanding the size of the dataset leads to improvements in performance. Islam N. et al. (19) employed data analysis-based solutions for IDSs because of their quick implementation and enhanced efficiency. Therefore, in our research, we employed Network Intrusion Detection Systems (NIDS) that rely on anomaly detection through the network extraction of data.
Training models, using a small data set presents significant challenges, such as (i) Overfitting is more likely to occur with a smaller dataset. (ii) Can introduce bias or increase model variance, which may reduce model accuracy and dependability. (iii) More vulnerable to noise and outlier effects. The training of the model may be significantly impacted by outliers, producing skewed results. (iv) Due to the limited parameter space exploration for a small dataset, determining the most suitable hyperparameter values becomes difficult. (v) It becomes challenging to train complex and deep neural networks using small datasets. So, we preferred to use the full version of UNSW-NB15 for our current experimental investigation rather than conducting our tests on a subset of the dataset because there have only been a few research

investigations carried out using the full version of the dataset. The incorporation of the complete version is necessary as the original dataset creators hadn't created separate sets for training and testing before the release of the complete version. As illustrated in Supplementary Table 1, it is evident that apart from the (16) (7), multiple experiments were carried out using subsets of the UNSW-NB15 dataset to enhance IDSs.
Overall, the following contributions have been made to the field of cyber security as the outcome of the current research study:

1. The current research makes a valuable contribution to the existing study of literature by presenting the enhanced efficacy of IDS in IoT systems through the integration of ensemble learning techniques, feature engineering, and normalization methods.
2. The current research study was designed to reduce the amount of time needed to create the models while maintaining the accuracy of ID on the full version of IoT IDS datasets.
3. Our proposed ensemble technique took advantage of the multiple base classifiers' strengths while minimizing their weaknesses, resulting in a more efficient and resilient IDS.
4. A real-time IDS is crucial for detecting intrusions in fast-connectivity systems. The current study identified ML classifiers exhibiting the highest detection accuracy along with the lowest model-building time for facilitating network IDS development.
5. The suggested approach incorporates the integration of FS techniques, especially the RF classifier’s feature relevance score. This classifier uses feature relevance ranking to determine the optimal subset of features, hence reducing training time and simplifying the IoT ecosystem.
6. The ensemble model that was developed consisted of multiple ML models, namely KNN, RF, LR, and SVM, serving as the base classifiers and XGBoost was employed as the meta-classifier. Our experimental investigation utilized two datasets: the UNSW-NB15 dataset, widely recognized as a benchmark for IDS performance, and the IoT Intrusion Dataset 2020 (IoTID20) dataset, specifically designed for studying Intrusion Detection (ID) in the context of IoT environments.
7. Due to the absence of experiments on complete versions of datasets, we utilized the FS technique on the complete dataset after the pre-processing step, resulting in compressed best feature sets that improved the performance of the IDSs.
8. Both binary, as well as multiclass categorizations of the two datasets, are taken into account to evaluate the suggested method.
9. False Positive Rate (FPR), Kappa Statistic (K), accuracy, precision, recall, f1-score Matthew's Correlation Coefficient (MCC), training time, and area under the receiver operating characteristic curve (AUC) value were used for evaluating the effectiveness of the proposed approach.
10. The experimental results demonstrate the efficacy of the proposed approach improves accuracy and minimizes the training times of IDS compared to other well-established supervised techniques outlined in the literature.

MATERIALS AND METHODS

The most important aspect of this study consists of the thorough evaluation and verification of IDS for the IoT. The evaluation is conducted by utilizing two widely recognized benchmarking IDS datasets, namely UNSW_NB15 and IoTID20. Both binary and multiclass evaluations were performed on the suggested model. The UNSW-NB15 dataset is renowned for its capacity to precisely represent network traffic, whereas the IoTID20 dataset was developed specifically to capture network traffic on IoT devices within a smart home setting. Several techniques, including anomaly detection, DL, and ensemble methods, have shown a remarkable ability to detect abnormal activity within IoT networks. To further improve the precision and effectiveness of ID in an IoT context, the current study attempts to emphasize the significance of feature engineering, selecting models with hyperparameter adjustment, and data reduction utilizing RF feature importance score. Figure 1 illustrates an architectural layout that outlines the several stages of the suggested model for the stacking framework.

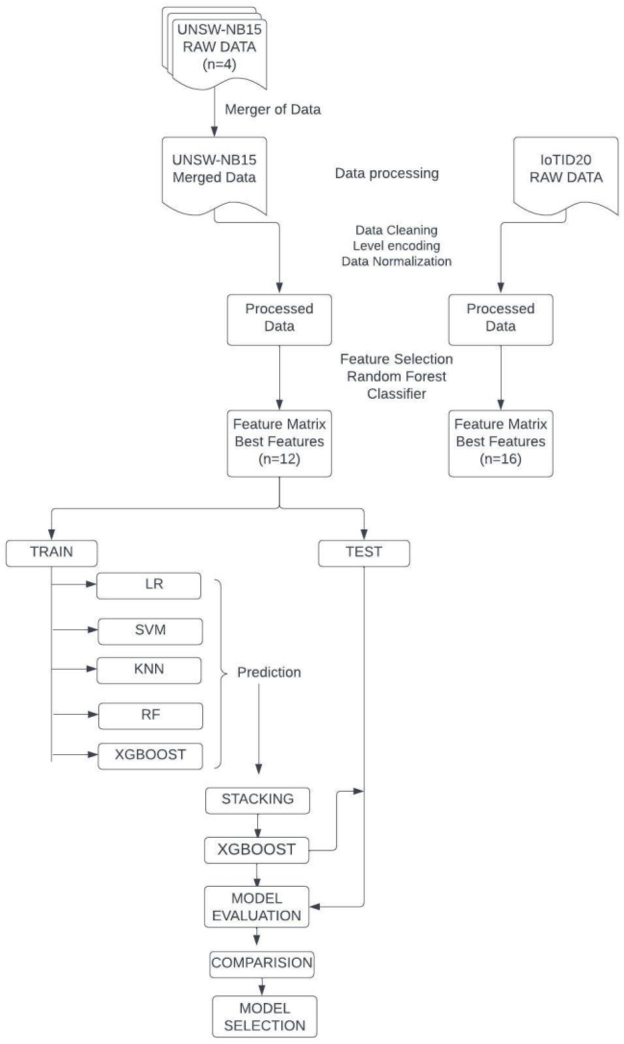

Figure 1. Ensemble-based framework for intrusion detection with feature engineering and normalization

The full version of the UNSW-NB15 dataset was divided into four separate Comma Separated Values (CSV) files as provided by the creator (10) (4). The integration of the four CSV files resulted in the creation of complete datasets. Following the merging process, pre-processing procedures were performed. The Python pandas package was utilized throughout the data pre-processing phase to get access to the CSV file. The analysis of the features' format and datatypes was conducted thoroughly before the pre-processing processes. Additionally, categorical characteristics were identified for label encoding. The data pre-processing phase consisted of several sub-phases, including data cleaning, label encoding, normalization and followed by an FS procedure that is based on the feature relevance score of the RF classifier. Through the use of random sampling, the dataset is divided into two subsets: the test set comprises 20% of the rows, and the training set comprises 80% of the rows. The complete training dataset was subsequently employed to train the primary classifiers, while the meta-classifier was used to generate predictions on the test data in the framework of stacking ensemble approaches. The XGBoost algorithm provides the capability to enhance the performance of a weak learner by evolving it into a strong learner, a process commonly referred to as boosting, which is achieved through its optimization stage. So, we used the XGBoost algorithm as a meta classifier, while LR, SVM, KNN, and RF are utilized as base classifiers. During the development of our research aimed at identifying the most efficient technique for IDS, we analyzed the outcomes produced by each different ML classifier. In our experimental analysis, we compared the predictive capability of classifiers with and without the FS method. The suggested framework was built using modules that are available within the Scikit-learn package (22). For assessing our models, we used the following metrics such as accuracy, precision, recall, F1-score, FPR, K, AUC value, training time, and predicting time.

2.1. Datasets Used in the Suggested IDS

The current section thoroughly explains the dataset utilized in the present research study. For experimental analysis, two different datasets were chosen. IoTID20 dataset which keeps track of network activities of IoT devices and another dataset namely UNSW-NB15 which provides detailed information about network traffic.

2.1.1 Description of IoTID20 Datasets

The IoTID20 dataset was created to improve the detection of cyberattacks targeting IoT networks (23). The creation of a smart home setting with two IoT devices—the SKT NGU and the EZVIZ Wi-Fi camera—produced the IoTID20 dataset (24). The devices in the IoT ecosystem that have been demonstrated as being susceptible to attack were the cameras with AI speakers. Among the many electronic gadgets linked to the network for smart homes are laptops, tablets, and smartphones, all of which can initiate attacks on other devices. The IoTID20 dataset has an overall 625,783 entries and contains a total of 83 network attributes in addition to three label attributes. The IoTID20 dataset includes labels namely anomaly (n=585710) and normal (n=40073) that are presented in binary classification formats. Tables 1 and 2 present a comprehensive analysis of the IoTID202 datasets in category-wise and sub-category-wise attack distribution in a multiclass classification scheme.

Table 1. Details relating to category-wise anomaly distribution of IoTID20.

Category	**Attribute**
Mirai	415677
Scan	75265
DoS	59391
Normal	40073
MITM ARP Spoofing	35377

Table 2. Details relating to subcategory-wise anomaly distribution of IoTID20.

Sub Category	**Attribute**
Mirai-UDP Flooding	183554
Mirai-Hostbruteforceg	121181
DoS-Synflooding	59391
Mirai-HTTP Flooding	55818
Mirai-Ackflooding	55124
Scan Port OS	53073
Normal	40073

MITM ARP Spoofing	35377
Scan Hostport	22192

2.1.2 Description of UNSW-NB15 Dataset

The UNSW-NB 15 dataset was created by using the IXIA Perfect Storm tool in the cyber range laboratory of the Australian Centre for Cyber Security (ACCS) (4). The dataset has been generated by a hybrid approach that incorporates real contemporary regular network traffic as well as artificially simulated modern attack activities. The present dataset comprises nine distinct categories of modern attack types and contains real events of regular network traffic that have been collected over some time. The data set includes nine distinct attack types namely Backdoors, Analysis, Denial of Service (DoS), Fuzzers, Exploits, Generic, Reconnaissance, Shellcode, and Worms. Twelve algorithms are developed using the Argus and Bro-IDS tools to generate a set of 49 features and the matching class label. The following subcategories have been created based on the features (10):

Flow features: include the identification attributes that are present between hosts and contain several features, including srcip, sport, dstip, dsport, and proto.

Basic features: contains the attributes that cover protocol connections are state, dur, sbytes, dbytes, sttl, dttl, sloss, dloss, service, sload, dload, spkts, and dpkts.

Content features: include TCP/IP attributes, as well as certain attributes specific to HTTP services, such as swin, dwin, stcpb, dtcpb, smeansz, dmeansz, trans_depth, and res_bdy_len

Time features: includes timing-related attributes including sjit, djit, stime, ltime, sintpkt, dintpkt, tcprtt, synack, ackdat, and round-trip time, as well as arrival times between packets and start/end packet times.

Additional generated features: each feature has its own set of security measures, and the included features are is_sm_ips_ports, ct_state_ttl, ct_flw_http_mthd, is_ftp_login, and ct_ftp_cmd. The features ct_srv_src, ct_srv_dst, ct_dst_ltm, ct_src_ ltm, ct_src_dport_ltm, ct_dst_sport_ltm, and ct_dst_src_ltm are constructed from the flow of 100 records' worth of connections in the order of the last time feature.

Label features: UNSW-NB15 datasets were labeled using two attributes: attack_cat, which stands for the nine categories of attacks as well as the normal category, and label, which is assigned a value of 0 for normal instances and 1 for all anomaly instances.

The dataset provided by (4) has been split into four CSV files denoted as UNSWNB15_1.csv, UNSW-NB15_2.csv, UNSW-NB15_3.csv, and UNSWNB15_4.csv. The records within each CSV file are organized by the latest time property value, from highest to lowest. The total amount of records among the four CSV files was 2,540,044. Based on the initial documentation, it has been reported that the first three CSV files have 700000 records each, while the fourth file contains 440044. A notable observation was made on the complete version of UNSW-NB15 during our experimentation three extra instances were found, resulting in a total of 2,540,047 instances. This figure is precisely the same as the amount that had been formerly validated by (7) and (16). Furthermore, the authors of the study generated a compact dataset that followed a pre-established distribution for training and testing. This dataset was labeled as UNSW_NB15_training-set.csv and UNSW_NB15_testing-set.csv.

Table 3 shows a concise summary of the overall count of instances contained inside the UNSW-NB15 dataset. Additionally, Table 4 illustrates the arrangement of 10 classes among the 4 CSV files.

Table 3. Summary of UNSW-NB15.

Name of Dataset	Filename	Number of records
Part of UNSW-NB15	UNSW_NB15_training-set.csv	175,341

	UNSW_NB15_testing-set.csv	82,232
Full UNSW-NB15	UNSW-NB15	2,540,047
	UNSW-NB15_1.csv	700,001
	UNSW-NB15_2.csv	700,001
	UNSW-NB15_3.csv	700,001
	UNSW-NB15_4.csv	440,044

Table 4. Arrangement of attacks in 4 CSV files of the UNSW-NB15 dataset.

Attack Name	**Name of the CSV file**			
	NB15_1	**NB15_2**	**NB15_3**	**NB15_4**
Normal	677786	647252	542576	351150
Generic	7522	27883	118198	61878
Exploits	5409	11103	16574	11439
Fuzzers	5051	4668	9137	5390
Reconnaissance	1759	4637	5642	4907
DoS	1167	3116	5582	3530
Backdoors	534	608	873	670
Analysis	526	370	759	666
Shellcode	223	324	593	371
Worms	24	40	67	43

2.2 Data Pre-Processing

This section presents an in-depth analysis of the datasets used in our experiment to identify their attributes, attack types, and data types. This investigation helps us to identify the important features that enhance the efficiency of ML methods. The pre-processing step plays an important role in preserving compatibility among our used IDS datasets and ML methods. The pre-processing of data includes important steps namely data cleaning, label encoding, feature scaling, and feature selection which will be discussed in subsequent sections. Python built-in package namely Pandas was used in the pre-processing step. The subsequent sections 2.2.4 and 2.2.5 provide a detailed description of further preparation techniques used on each specific data set.

2.2.1 Label Encoding

Since certain ML algorithms are not capable of processing categorical characteristics, it is necessary to convert these features into a numerical format before their utilization. Label encoding is a well-known encoding strategy for dealing with categorical values that assign each categorical value a distinct numeric value. Due to the high cardinality of the categorical features, using the technique of one-hot encoding leads to an increase in data dimensionality. Consequently, this results in an increase in memory space requirements and time required for computation, as noted by (25). Therefore, in current research work, the label encoder strategy was applied to convert the categorical data into numeric because the datasets used in our experimental analysis, namely the UNSW-NB15 and IoTID20 datasets have several categorical features.

2.2.2 Feature Selection (FS)

Certain ML techniques exhibit high performance in IoT IDSs; however, the process of model creation becomes notably time-consuming when dealing with huge datasets. Therefore, a comprehensive feature analysis was required to determine the distinctive characteristics of IoT networks. Our study utilized two datasets, namely IoTID20 and UNSW-NB15, which consist of a comprehensive set of attributes pertaining to network traffic. It is imperative to exclude extraneous attributes from datasets that do not contribute to the output label, as they might lead to overfitting and underfitting, hence significantly affecting the performance and execution time of the classifier. Since the most important features are chosen using the RF classifiers' highest priority scores, it becomes simpler to identify intricate patterns in the network traffic data from both datasets. Chen RC. et al. (26) further examined the benefits of FS in their study. Their findings indicate that FS techniques are effective in reducing the number of parameters, thereby minimizing training time, mitigating overfitting through improved generalization, and addressing the challenges posed by high-dimensional data, as well as avoiding the dimensionality curse. The researchers also employed the RF algorithm as an FS method to exclude irrelevant characteristics. The experimental analysis conducted in their study provides compelling evidence that the RF algorithm functioned effectively as an FS strategy, resulting in enhanced performance of the ML-based model. RF method is a robust ML technique that exhibits the capacity to do FS, while effectively tackling challenges related to classification and regression tasks. Our main focus is to utilize the feature importance score of the RF method to find a subset of appropriate features from the initial collection to enhance the predictive ability of ML models and reduce the issue of overfitting. Accordingly, we went through several experiments to remove the lower feature importance score for finding out the best feature set which helps the classification model that would not only improve the detection accuracy but also minimize the model building time. Out of the complete set of pre-processed data, we select 12 features from the UNSW-NB15 dataset and 16 features from the IoTID20 dataset using an RF classifier's feature relevance ranking. The feature names of the IoTID20 dataset utilized in our tests are presented in Table 5 of section 2.2.4, while the feature names of the UNSW-NB15 dataset are shown in Table 6 of section 2.2.5.

2.2.3 Feature Normalization

The learning process of ML methods, such as SVM, LR, ANN, and KNN, is influenced by the presence of large numerical values in the datasets. Furthermore, the training of datasets with high dimensions requires a significant utilization of computational resources (9). There exist several normalization techniques, such as standardized moment, z-score normalization, and min-max normalization (27). Feature normalization, which is often referred to as feature scaling or data normalization, is a preprocessing method employed in the field of ML. The objective of this is to ensure that the features incorporated in a model exhibit uniform scales. The act of normalizing features has the potential to enhance the performance of numerous ML algorithms and provide stability within the optimization process. The UNSW-NB15 and IoTID20 datasets exhibit variations in the values of their features. The min-max normalization approach was employed to normalize the data within the range of 0 and 1 using the min-max method, as shown by the following equation:

$$\mathrm{X_{norm}} \; \frac{\mathrm{X} - \mathrm{X_{min}}}{\mathrm{X_{max}} - \mathrm{X_{min}}}$$

Here, $\mathrm{X_{min}}$ and $\mathrm{X_{max}}$ Represents the lower and upper bounds of attribute X. The variable, $\mathrm{X_{norm}}$ Represents the normalized value of the attribute X, while X denotes the original value of the feature.

2.2.4 IoTID20 Dataset Pre-processing

During the pre-processing step, duplicate features are removed first, and then NaN values are used in place of INF values. The median value of the feature is utilized to replace missing values if the percentage of empty values for that feature exceeds 5%. Because label encoding has advantages over one-hot encoding in terms of memory usage and computational performance, it is recommended for categorical features like Src Port, Dst Port, and Protocol. The best 16 pertinent features were chosen after the FS approach was applied to the pre-processed data, and their feature importance score is shown in Table 5.

Table 5. 16 Features selected using RF Classifier from the full version of IoTID20 datasets.

Features name	Feature importance score
Dst_Port	0.200052
Flow_Duration	0.076488
Src_Port	0.073024
Init_Bwd_Win_Byts	0.057825
Flow_Pkts/s	0.046540
ACK_Flag_Cnt	0.033909
Flow_IAT_Max	0.032497
Bwd_Pkts/s	0.030386
Bwd_IAT_Tot	0.028496
Flow_IAT_Std	0.028162
Bwd_Header_Len	0.026049
Idle_Mean	0.025996
Idle_Max	0.025756
Flow_IAT_Mean	0.024333
Fwd_Pkt_Len_Std	0.022513
Idle_Std	0.014622

2.2.5 UNSW-NB15 Dataset Pre-Processing

The relevant headers were added to each of the four data frames after importing the CSV files into Panda's data frame. Subsequently, the four distinct data frames were merged into a single data frame to generate a complete dataset. The samples with duplicate features are removed and the "-" values are replaced with NaN values using the NumPy library. The attribute's median value is used to fill in the missing values when the percentage of missing values for that attribute exceeds 5%. Categorical attributes, such as proto, state, and service, are encoded using a label encoder. Then, the pre-processed data was used for the selection of the top 12 pertinent features through the RF feature importance ranking score. These features, together with their corresponding feature importance scores, are presented in Table 6.

Table 6. 12 Features selected using RF Classifier from the full version of UNSW-NB15 datasets.

Features name	Feature importance score
ct_state_ttl	0.182644
sttl	0.166886
ackdat	0.068337
Sload	0.049987
dttl	0.049946
dmeansz	0.045358
tcprtt	0.044858
synack	0.040415
sbytes	0.038859
smeansz	0.035734
dbytes	0.030914
Dload	0.021711

2.3 Overview of ML Algorithms

2.3.1 Extreme Gradient Boosting (XGBoost)

XGBoost distinguishes itself from other libraries by offering the capability to incorporate and optimize regularization configurations. The method demonstrates a high level of effectiveness in minimizing computational time and optimizing the utilization of memory resources. XGBoost is a highly optimized distributed gradient boosting library that has been designed to build reliable ML models (28). XGBoost to improve their models XGBoost has been widely used in several Kaggle contests by the winning team to enhance their models (29).

2.3.2 K-Nearest Neighbor (KNN)

KNN is a supervised ML method that uses the Euclidean distance function to find out the closeness or difference between the two features in a dataset (12). The Euclidean distance among two points represented as s (a, b), can be calculated using the following mathematical equation:

$$s(a,b) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2}$$

Where a_i denotes the ith feature of class a, whereas b_i denotes the ith feature of class b and "n" denotes the total number of features in the dataset.

2.3.3 Support Vector Machine (SVM)

SVM is a well-known supervised ML algorithm that uses a hyperplane to distinguish between negative and positive class variables by minimizing the structural risk(27). Radial Basis Function (RBF) is a widely used kernel function to enhance the model accuracy (12). The primary objective of the current research study was to improve the accuracy of the models by using RBF of SVM and the equation is defined as follows:

$$(a,b) = \exp\left(-\frac{||a-b'||^2}{2\sigma^2}\right)$$

where $||a-b'||^2$ represents the squared Euclidean distance between two data points a and b.

2.3.4 Random Forest (RF)

RF which is a supervised ML method creating a forest by many DT. The effectiveness of individual trees is improved by integrating bootstrap aggregating and selection of nodes during the creation of a DT by randomization (26).

2.3.5 Logistic Regression (LR)

LR is an ML method that uses probability concepts to make predictions and is used in both binary and multiclass classification problems (9). As LR has high computational efficiency and minimal resource demands it effectively handles large datasets (11). The cost function used in LR is known as a sigmoid function which is used to assign a unique value between 0 and 1 to each real number. This technique is employed to establish a correspondence between predictions and their associated probabilities. The hypothesis expectation of LR is illustrated as follows:

$0 \leq h0(x) \leq 1$

2.4 Performance Evaluation

The primary aim of our work is to optimize the accuracy of predictions for samples inside the test dataset, despite the presence of multiple performance measurements available for evaluating ML-based IDSs. So, to achieve the optimized accuracy, the models were trained by using all features and selected reduced features from both datasets. To determine the effectiveness of the suggested approach a wide range of evaluation metrics were used. The metrics considered in this study include accuracy, recall, kappa statistics (K), Mathew Correlation Coefficient (MCC), Area Under the Receiver Operating Characteristic Curve (AUC-ROC), false positive rate (FPR), precision, F1-score, and the duration of model training. Several different types of assessment metrics can be derived from the confusion matrix, which is a visual representation of the data being analyzed. The four values associated with it are as follows: False Negative (FN), True Negative (TN), False Positive (FP), and True Positive (TP).

- **TP:** When an IDS identifies an occurrence as an attack, it signifies the detection of an actual attack.
- **FP:** When an instance is categorized as an attack by the IDS, although the instance is later confirmed to be normal.
- **TN:** When the IDS identifies a scenario as normal and further investigation reveals that the instance is, in fact, normal.
- **FN:** IDS made a mistake in classifying the instances as normal while they are, in fact, malicious attacks.
- **Accuracy:** The accuracy of a prediction system can be expressed as the fraction of predictions that turned out to be accurate out of the total number of predictions.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

The usefulness of accuracy as a performance metric becomes clear when the distribution of the target class is balanced, but it may not be the optimal choice for imbalanced classes. Relying just on accuracy as the evaluation metric for an ML model without doing a comprehensive assessment utilizing other evaluation metrics can result in issues when deploying the model on unseen data, perhaps leading to unsatisfactory predictions.

- **Precision:** The precision metric measures the prediction accuracy of a model, specifically the proportion of accurately predicted truly positive cases.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- **Recall:** Recall, detection rate, or sensitivity refers to the capacity of a model to effectively recognize and categorize attacks. The recall for a given label can be mathematically described as the proportion of true positives to the total quantity of genuine positives.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **F1- Score:** The F-measure is a metric utilized for assessing the effectiveness of binary classification algorithms. The F1-score has been calculated by taking the harmonic mean of both accuracy and recall. The maximum value is achieved when precision is equivalent to the recall. F1-score is calculated by taking the harmonic mean of recall and precision. The larger value of the F1-score is achieved recall equal to precision.

$$\text{F1} = 2.\frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

- **Kappa Statistic(K):** By using the concept of probability statisticians use kappa statistics to assess the correlation between expected accuracy and observed accuracy. The correlation coefficient equals 0 when there is no correlation between real and predicted values and the correlation value equals to1 when there is a strong correlation.
- **Matthews Correlation Coefficient (MCC):** In case of data is not balanced MCC is a useful metric for assessing the effectiveness of binary classification models. MCC values are in the range of -1 to +1 whereas +1 represents high accuracy whereas -1 represents low accuracy in all predictions. Most benchmark IDS datasets are not balanced which means

they have a larger number of attacks in comparison with normal logs. Therefore MCC is frequently used as an evaluation metric (18).

$$\text{MCC} = \frac{\text{TP} * \text{TN} - \text{FP} * \text{FN}}{\sqrt{((\text{TP} + \text{FN})(\text{TP} + \text{FP})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}}$$

- **ROC-AUC (Area Under the Curve of the Receiver Operating Characteristic):** AUC is a numerical metric that measures the size of the area under the ROC curve. ROC curve is a commonly used evaluation metric in binary classification problems. It represents a short description of the ROC curve. ROC curve is a graphical representation that indicates the relationship between the model’s ability to correctly identify instances(recall) and incorrectly classify negative instances as False Positive Rate (FPR). FPR can be determined by using the following mathematical equation.

$$\text{FPR} = \frac{\text{FP}}{\text{TN} + \text{FP}}$$

- **Training Time**: The duration required for training ML models with a specific dataset is quantified in units of time, namely seconds (s). This duration is obtained by calculating the time gap between the initiation and completion of the training process.
- **Predicting Time:** The duration required for ML models to make predictions on the test data of a particular dataset was determined by calculating the difference between the end time of the training process and the end time of the prediction process.

RESULTS AND DISCUSSION

Different supervised ML models and stacking-based ensemble learning algorithms are used in different contexts. These models are evaluated using a variety of evaluation metrics on benchmarking datasets for IDSs. Furthermore, we have lowered the feature count to enhance the IDSs and contrasted the results with and without FS to assess the influence of FS. We utilized the complete versions of two benchmarking IoT IDS datasets, specifically the full version of UNSW-NB15 (4) and IoTID20 (24) to assess the effectiveness of our suggested model. Two distinct datasets of different sizes are used in the experimental research to evaluate the effectiveness of our suggested methodology. The UNSW-NB15 dataset is significantly larger as compared to the IoTID20 dataset. Based on the outcomes of our experimental investigations, it has been found that our proposed methodology exhibits satisfactory performance in both datasets. The experiments using the UNSW-NB15 dataset are performed on a computer system operating on a 64-bit Microsoft Windows 10 platform with an Intel(R) Core (TM) i7-3777OS CPU running at a clock speed of 3.10 GHz (3.09 GHz) and had a total memory capacity of 16 GB. Experiments on IoTID20 are carried out on a system operating Microsoft Windows 11 along with an AMD Ryzen 7 3700U processor, a 2.30 GHz Radeon Vega Mobile Gfx graphics card, and 8 GB of RAM. The ML models are developed, prepared, evaluated, and verified utilizing the Scikit-Learn Python framework (22). In addition to offering creative applications of several well-known ML algorithms, Scikit-learn guarantees an intuitive user interface that seamlessly integrates with the Python programming language. The Pandas and NumPy libraries were utilized in the pre-processing stages to import the CSV files, filter, encode, and normalize the data. Also, the Matplotlib library was utilized for data visualization.

The experimental design consisted of three stages, in which we used LR, KNN, SVM, and RF as base classifiers and XGBoost as the metaclassifier. During the first stage, the configurations of each ML method are optimized which has been outlined in Table 7. In the second stage of the design, we utilized all the network features, including 47 features obtained from the UNSW-NB15 dataset and 83 features obtained from the IoTID20 dataset. In the third stage, a feature vector having fewer features was created by utilizing the feature importance score of the RF method. This could reduce the feature vector by having only 12 features from the UNSW-NB15 dataset and 16 features from the IoTID20 dataset. The analysis was conducted by applying the reduced feature vector, which included both the binary as well as multiclass classification approaches. Section 3.1 analyses the critical hyperparameter values for the UNSW-NB15 and IoTID20 datasets using XGBoost, RF, LR, SVM, and KNN methods. The outcomes obtained from the optimization of hyperparameters in our research investigation on IoTID20 are discussed in section 3.2 and the analyses of UNSW-NB15 are discussed in section 3.3 respectively.

3.1 Parameter Optimization

While there are several ways to assess ML-based IDSs, our study is primarily concerned with maximizing the correct predictions on the test datasets following model training with the training data. From the literature survey, we found that most of the ID models did not consider hyperparameter tuning of ML methods. Our focus is to improve the model performance on unseen data. To achieve this goal, we optimized the hyperparameters of ML algorithms.

Hyperparameter tuning refers to the systematic exploration of various combinations of hyperparameters intending to improve the performance of an ML model. It is a crucial step in the ML model development process. Selecting the right hyperparameters has a big impact on how well the model extends from training data to unseen data. We conducted a series of experiments on two datasets that are used in our experimental analysis to evaluate the impact of hyperparameter tuning on the performance of LR, SVM, KNN, RF, and XGBoost. The used algorithms have many hyperparameters that can be adjusted. We optimized the hyperparameters of each algorithm to improve the effectiveness of detection in IoT IDSs. The models are passed through rigorous testing, wherein different hyperparameters are systematically adjusted. After careful analysis of the results, we have identified the optimal hyperparameters that yield the most desirable results which are comprehensively provided in Table 7.

Table 7.Critical hyperparameter values for performance evaluation of five ML algorithms on UNSW NB15 and IoTID20 datasets.

Model	UNSW-NB15	IoTID20
XGBoost	Learning rate: 0.3	Learning rate: 3
	Max_depth: 9	Max_depth: 9
	Estimators: 600	Estimators: 600
RF	Estimators: 200	Estimators:200
	Min_samples_split: 2	Min_samples_split: 2
		Max-depth: 30
		Criterion: Gini
LR	C: 100	C: 100
	Solver: Liblinear	Solver: Liblinear
	Max-iter:200	Max-iter: 200
	Penalty: L1	Penalty: L2
KNN	N_neighbors: 7	N_neighbors: 7
	Leaf_size: 30	Leaf_size: 30
	Metric: Minkowski	Metric: Minkowski
	P: 1	P: 1
SVM	Kernel: RBF	Kernel: RBF
	C: 1000	C: 1000
	Gamma: 0.6	Gamma: 0.6

During the process of modeling, the parameters specified in Table 7 are adjusted to optimize the efficacy of the ML model. Only the learning rate, maximum depth of trees, and several boosting rounds (estimators) are optimized for the XGBoost algorithm; all other parameters are left at their default values. The XGBoost classifier has been selected as the metaclassifier in our model because of its potential to mitigate both variation and bias. Also, the XGBoost algorithm utilizes the max-depth parameter as a designated criterion for tree pruning, resulting in a substantial enhancement of performance. Using the variation of 'learning rate' as 0.1, 0.2, 0.3, and 0.4, XGBoost produces an accuracy of 99.84%, 99.89%, 99.91%, 99.92% with a time variation of 30.42 s, 34.74 s,30.03 s and 30.59 s respectively and also FPR was reduced from 0.016 to 0.008 in IoTID20 dataset. XGBoost learning rate, max_depth, and estimators are also tuned in the combination of (0.1,3,200) and (0.3,7,1000), and accuracy increased from 99.84% to 99.94%, FPR decreased from 0.0162 to 0.0061 in IoTID20 binary classification. In the case of UNSW-NB15 binary classification, the hyperparameters of XGBoost, namely the learning rate, max_depth, and estimators, were adjusted using two combinations: (0.1, 3, 200) and (0.3, 9, 600). As a result, the accuracy was improved from 99.13% to 99.29%, while the FPR was reduced from 0.0040 to 0.0036. The results of the experimental analysis of the best models for two datasets are discussed in the next sections.
We also optimized the hyperparameters of base classifiers LR, RF, SVM, and KNN in UNSW-NB15 and IoTID20 datasets which are shown in Table 7. As for RF on the IoTID20 dataset, the number of trees in the forest(estimators) increased from 100 to 200, and the accuracy increased from 99.77% to 99.78%. In the case of SVM, when the regularization parameter 'C' increased from 100 to 1000, the accuracy increased from 99.20% to 99.40%, the FPR reduced to 0.0584 from 0.0748 and training time was increased from 969.68 s to 1211.56 s in IoTID20 dataset. In the case of LR, when the regulation hyperparameter changed to L2 from L1 the FPR decreased to 0.1944 from 0.2188 in the IoTID20 dataset. To lower the FPR and training time, the number of neighbors hyperparameter value of KNN was also optimized.

3.2 Results using IoTID20 Dataset

It is challenging for the current ML-based IDS to enhance its capability to precisely detect different types of cyberattacks. The objective of our proposal is to enhance performance and minimize computational resources required for the identification of cyberattacks. Tables 8a, 8b, 9a, 9b,10a, and 10b presented the best results obtained by applying five different ML algorithms and stacking-based ensemble method on the IoTID20 dataset's full feature set, and reduced feature set for binary classification, category-wise multiclassification, and subcategory-wise multiclassification results. Figures. 2, 3, and 4 depict the confusion matrices using the decreased feature set for both binary and multiclass attack detection schemes of the IoTID20 dataset. The confusion matrices demonstrated that the ML models correctly identified most of the classes and had the lowest FPR in our proposed FS approach.
Table 8a. IoTID20 binary classification study utilizing all features.

ML Method	XGBoost	RF	SVM	LR	KNN	Ensemble
Test AC	99.94	99.78	99.40	97.31	99.70	99.83
Recall	99.94	99.78	99.40	97.31	99.70	99.83
Precision	99.94	99.78	99.40	97.23	99.70	99.83
F1	99.94	99.78	99.39	97.24	99.70	99.83
FPR	0.0063	0.0212	0.0584	0.1944	0.0254	0.0141
MCC	99.62	98.71	96.48	83.74	98.22	99.03
K	99.62	98.70	96.44	83.59	98.22	99.03
AUC	99.68	98.92	97.05	89.81	98.69	99.27
Training time	287.22	48.72	1211.56	97.79	0.06	13344

Predicting Time	0.55	0.75	65.29	0.02	2525.2	822.66
Total time	287.77	0.49	1276.85	97.8	2525.2	14166

Table 8b. IoTID20 binary classification study utilizing reduced features.

ML Method	**XGBoost**	**RF**	**SVM**	**LR**	**KNN**	**Ensemble**
Test AC	99.94	99.84	99.14	96.89	99.81	99.88
Recall	99.94	99.84	99.14	96.89	99.81	99.88
Precision	99.94	99.84	99.14	96.78	99.81	99.88
F1	99.94	99.84	99.12	96.79	99.81	99.88
FPR	0.0057	0.0157	0.0806	0.2279	0.0174	0.0105
MCC	99.63	99.05	94.90	81.05	98.89	99.29
K	99.63	99.04	94.82	80.82	98.89	99.29
AUC	99.71	99.20	95.91	88.08	99.12	99.46
Training time	111.38	40.04	541.63	10.71	0.03	8655.83
Predicting Time	0.41	0.72	62.91	0.02	684.55	694.89
Total time	111.79	40.76	604.54	10.72	684.59	9350.72

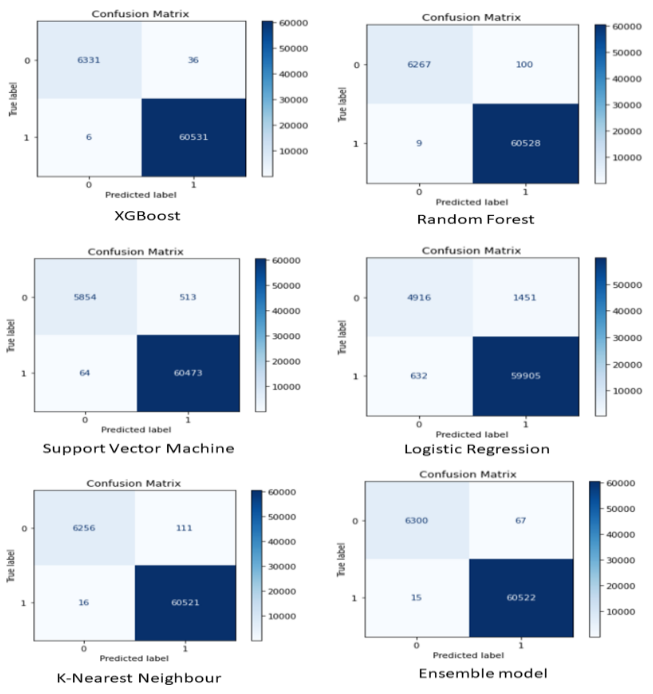

Figure 2. Confusion matrix of IoTID20 binary classification (16 features)

The results of the comparison between Tables 8a with 8b and Figure 2 indicate that ensemble learning outperforms SVM, RF, LR, and KNN algorithms. The ensemble learning strategy achieved an accuracy of 99.88% with an FPR of 0.0105 by just utilizing 16 network features. Also, the training time was decreased from 133,344 s to 8,655.83 s while employing the FS technique in the ensemble method. When it came to accuracy and FPR, LR produced the lowest performance results when compared to other classifiers, while XGBoost performed better. However, when it comes to predicting time, LR outperforms other classifiers. Based on the findings outlined in Tables 8a and 8b, it can be observed that the KNN algorithm exhibits a comparatively shorter duration for training, but its prediction time surpasses that of other ML methodologies. The training duration of SVM is slightly longer in comparison to other classifiers. For binary attack detection, the SVM method using the RBF as its kernel obtained test accuracy of 99.40% and 99.14%, respectively, using the full and reduced feature sets. Table

8b also illustrates that the using reduced features, the RF classifier's performance was satisfactory in comparison to other classifiers with an accuracy of 99.84%. In addition, it had a low FPR of 0.0157, completed the training phase in 40.04 s, and completed the prediction phase in just 0.72 s. Using 16 features, the XGBoost model produced a test accuracy of 99.94% with an FPR of 0.0057.
Table 8b's analysis indicates that the FPRs of the ML techniques decreased using a smaller feature set. For example, XGBoost's FPR decreased from 0.0063 to 0.0057, RF's FPR decreased from 0.0212 to 0.0157, KNN's FPR decreased from 0.0254 to 0.0174, and the ensemble learning approach's FPR decreased from 0.0141 to 0.0105. Additionally, using the reduced feature set, the training times of several classifiers dropped as well: in XGBoost, from 287.22 s to 111.38 s in RF, from 48.72 s to 40.04 s in SVM, from 1211.56 s to 541.63 s in LR, from 97.79 s to 10.71 s in KNN, from 0.06 s to 0.03 s; and in the ensemble learning technique, from 13344 s to 8655.83 s. The results displayed in Table 8b demonstrate our investigation's conclusions—that the IoTID20 binary attack identification approach's use of a smaller feature set led to a significant drop in the FPR. Also, the training and prediction times were further improved by the use of the smaller feature set.

Table 9a. IoTID20 multiclass classification(category) study utilizing all features.

ML Method	**XGBoost**	**RF**	**SVM**	**LR**	**KNN**	**Ensemble**
Test AC	99.58	96.7	93.93	86.13	97.65	97.86
Recall	99.58	96.7	93.93	86.13	97.65	97.86
Precision	99.58	96.64	93.82	83.92	97.64	97.84
F1	9958	96.66	93.49	82.4	97.64	97.85
FPR	0.006	0.019	0.025	0.071	0.021	0.0151
MCC	99.28	94.32	89.49	75.4	95.96	96.32
K	99.28	94.31	89.35	73.74	95.96	96.32
Training time	2246.29	58.17	5964.01	366.94	0.16	40732.7
Prediction Time	5.71	1.3	583.18	0.13	4883.9	1366.95
Total time	2252	59.47	6547.18	367.07	4884.1	42099.7

Table 9b. IoTID20 multiclass classification(category) study utilizing reduced features.

ML Method	**XGBoost**	**RF**	**SVM**	**LR**	**KNN**	**Ensemble**
Test AC	99.64	98.22	91.49	85.65	98.29	98.65
Recall	99.64	98.22	91.49	85.65	98.29	98.65
Precision	99.64	98.21	91.8	79.56	98.29	98.64
F1	99.64	98.21	89.87	81.06	98.29	98.64
FPR	0.004	0.011	0.032	0.068	0.013	0.012
MCC	99.38	96.95	85.32	74.6	97.07	97.68
K	99.38	96.94	84.97	72.48	97.07	97.68
Training time	695.41	40.63	4201.68	53.11	0.03	33314.7
Prediction Time	2.79	1.1	555.01	0.02	949.39	1272.63
Total time	698.2	41.74	475.69	53.12	949.43	34587.3

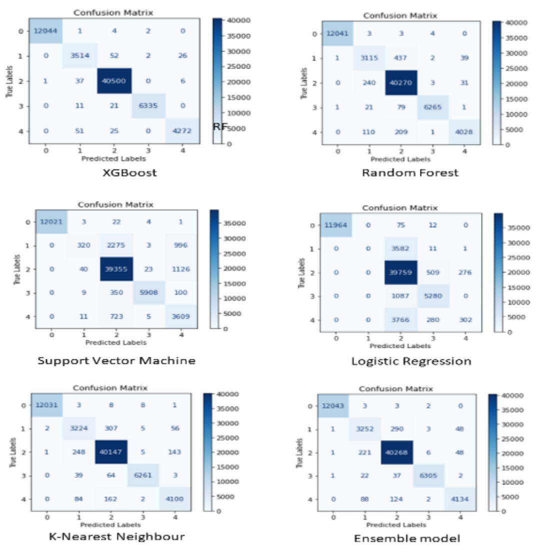

Figure 3. Confusion matrix of IoTID20 category-wise multiclassification (16 features)

Tables 9a and 9b present the average accuracy, FPR, MCC, kappa statistics, precision, recall, training time, and F-score of various ML methods used for the analysis of attack categories in the IoTID20 dataset. The results shown in Table 9b indicate that the test accuracies of the ML techniques used in this research study were improved for the category-wise attack detection multiclass classification scheme in the IoTID20 dataset. This improvement was achieved by reducing the number of network features from 83 to 16 through multiple iterations of hyperparameter tuning. More specifically, XGBoost accuracy improved from 99.58% to 99.64%, RF's accuracy improved from 96.7% to 98.22%, KNN's accuracy improved from 97.65% to 98.29%, and stacking-based ensemble methods' accuracy improved from 97.86% to 98.65%. With smaller feature sets, neither the LR nor the SVM could raise the performance metrics of the multiclass classification scheme. The results of Table 9b and Figure 3 indicate that the FPRs of XGBoost, RF, LR, KNN, and ensemble approaches are much lower when 16 features out of 83 network features are used. Also, Table 9b demonstrates that training and prediction times have decreased in all scenarios where ML approaches are used with FS to detect multiclass attacks.

Table 10a. IoTID20 multiclass classification to identify sub-categories of attacks utilizing all features.

ML Method	**XGBoost**	**RF**	**SVM**	**LR**	**KNN**	**Ensemble**
Test AC	85.85	79.6	80.24	73.42	80.54	92.95
Recall	85.85	79.6	80.24	73.42	80.54	92.95
Precision	85.04	79.19	79.53	70.87	83.33	92.89
F1	85.29	79.33	78.97	68.65	81.85	92.84
FPR	0.018	0.033	0.033	0.04	0.03	0.011
MCC	82.46	74.7	75.48	67.31	76.25	91.27
K	82.43	74.69	75.34	66.13	76.16	91.26
Training time	366.07	50.85	5526.74	961.51	0.06	32558.1
Prediction Time	0.43	1.75	1172.86	0.18	2438	1875.69
Total time	366.51	52.6	6699.6	961.69	2438.05	34433.8

Table 10b. IoTID20 multiclass classification to identify sub-categories of attacks utilizing reduced features.

ML Method	XGBoost	RF	SVM	LR	KNN	Ensemble
Test AC	86.07	81.79	78.30	72.14	82.01	92.66
Recall	86.07	81.79	78.30	72.14	82.01	92.66
Precision	84.59	81.34	77.82	62.61	84.15	92.53
F1	85.1	81.5	74.75	65.12	83.01	92.51
FPR	0.019	0.025	0.032	0.038	0.027	0.01
MCC	82.67	77.42	7307	65.89	78	90.9
K	82.6	77.41	7253	64.12	77.93	90.89
Training time	145.8	42.98	3436.37	89.02	0.04	24901.1
Prediction Time	0.63	1.97	798.32	0.02	673.58	1412.17
Total time	146.43	44.95	4234.69	89.05	673.62	26313.3

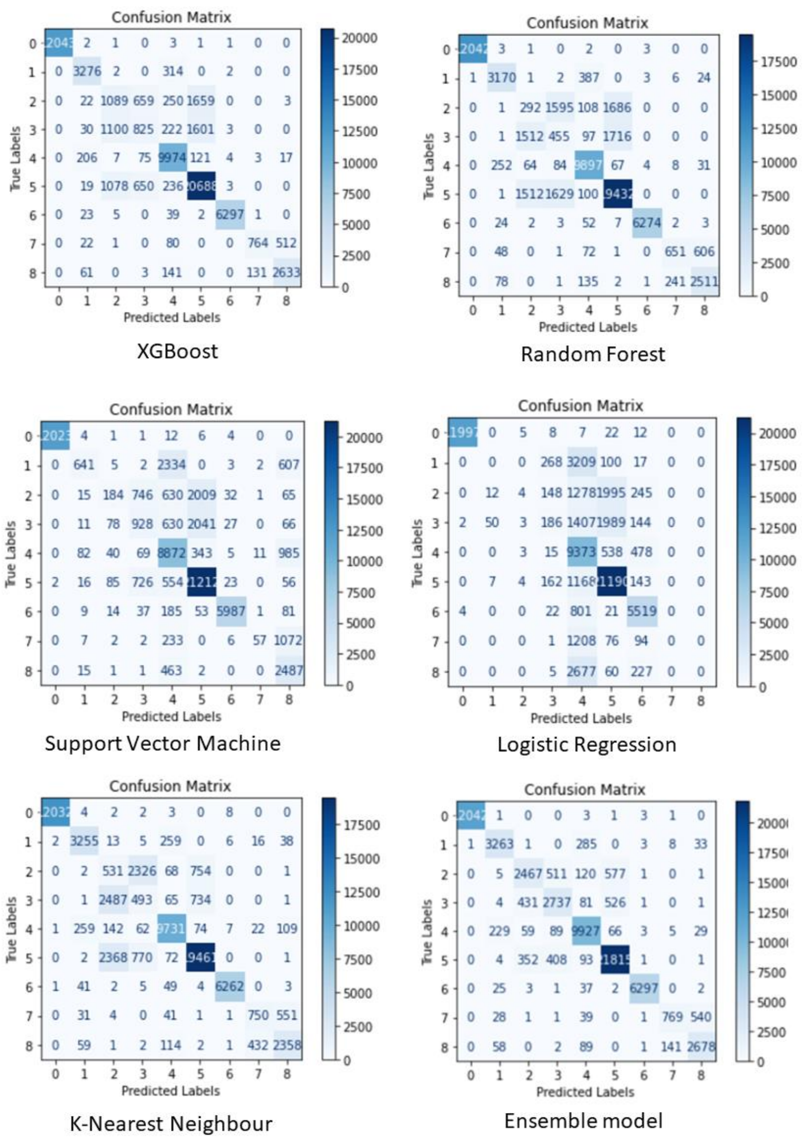

Figure 4. Confusion matrix of IoTID20 sub-category-wise multiclassification (16 features)

The average accuracy, FPR, MCC, precision, recall, training, F-score, and kappa statistics of the several techniques employed for the attack analysis in the sub-categories scheme of the

IoTID20 dataset are shown in Tables 10a and 10b. According to Table 10b and Figure 4, most ML techniques employed in this study were able to identify attacks inside the sub-category of a multiclass classification scheme with greater accuracy when the number of network features was reduced from 83 to 16. With smaller feature sets, the performance of ML methods such as XGBoost, RF, KNN, and ensemble techniques improved. On the other hand, there was no improvement in the accuracy of the LR and SVM models when using smaller feature sets. However, as Table 10b shows, the training times for LR and SVM were significantly less when using a reduced feature set as opposed to the full set of features. The statistics in Table 10b demonstrate that for the majority of ML approaches, employing smaller feature sets leads to a noticeable reduction in FPRs and training time. Using a smaller feature set, the stacking base ensemble approach achieves an accuracy rate of 92.66% and an FPR of 0.01 for the subcategory attack detection of IoTID20.

To summarize, the proposed method with the FS technique, which is shown in Tables 9b and 10b, shows a substantial reduction in the FPR and a shorter training time for multiple ML algorithms in the multiclass classification attack detection scheme of the IoTID20 dataset.

3.3 Results Using UNSW-NB15 Dataset

This section presents an analysis of the results obtained from the identification of attacks using both binary and multiclass classification schemes on the complete UNSW-NB15 dataset. Multiple experiments were undertaken in our analysis to enhance the detection accuracy of ML approaches by making adjustments to hyperparameters. The UNSW-NB15 dataset's most effective models utilizing all features and the smaller set of features generated via ML approaches and stacking ensemble methods are displayed in Tables 11a,11b,12a, and 12b, respectively. Figures. 5 and 6 display the confusion matrices of our analysis using the reduced set of characteristics. The suggested model had the lowest FPR and could accurately identify the majority of the classes, as shown by the confusion matrices.

Table 11a. UNSW-NB15 binary classification study utilizing all features.

ML Method	**XGBoost**	**RF**	**SVM**	**LR**	**KNN**	**Ensemble**
Test AC	99.29	99.30	99.07	98.66	98.98	99.29
Recall	99.29	99.30	99.07	98.66	98.98	99.29
Precision	99.29	99.30	99.08	98.76	98.98	99.29
F1	99.29	99.30	99.07	98.69	98.98	99.29
FPR	0.0036	0.0034	0.0055	0.0102	0.0051	0.0032
MCC	92.26	92.41	89.98	86.36	88.91	92.27
K	92.26	92.41	89.97	86.21	88.90	92.27
AUC	96.00	95.96	95.46	95.59	94.21	95.72
Training time	103.91	186.17	23004	1525.25	0.42	56959.96
Prediction Time	2.03	1.98	2357.22	0.05	6928.52	5042.35
Total time	105.94	188.16	25361.22	1525.29	6928.95	62002.31

Table 11b. UNSW-NB15 binary classification study utilizing reduced features.

ML Method	**XGBoost**	**RF**	**SVM**	**LR**	**KNN**	**Ensemble**
Test AC	99.04	99.14	98.57	97.91	98.96	99.13
Recall	99.04	99.14	98.57	97.91	98.96	99.13
Precision	99.04	99.14	98.89	98.11	98.96	99.13
F1	99.04	99.14	98.65	97.99	98.96	99.13
FPR	0.0048	0.0041	0.015	0.0151	0.0055	0.0045

MCC	89.53	90.62	87.16	79.22	88.74	90.52
K	89.53	90.62	86.36	78.98	88.74	90.52
AUC	94.56	95.00	99.19	92.51	94.38	95.18
Training time	59.62	101.94	10444.23	11.57	14.47	88782.29
Prediction Time	1.83	1.69	2730.77	0.03	268.24	3005.35
Total time	61.45	103.62	13174.99	11.61	282.70	91787.64

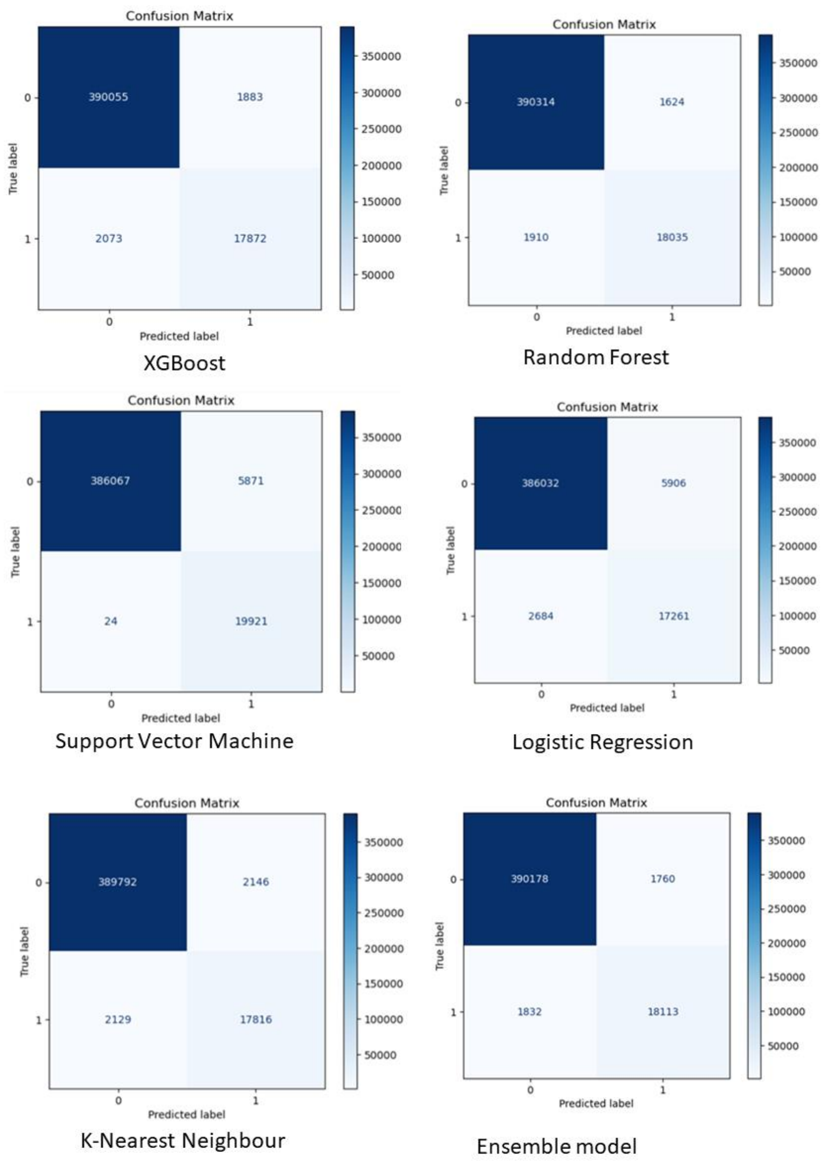

Figure 5. Confusion matrix of UNSW-NB15 binary classification (12 features)

According to the data presented in Table 11b and Figure 5, the results of our research demonstrate that ensemble learning outperforms the majority of ML-based methodologies. The ensemble model demonstrated a test accuracy of 99.13%, an FPR of 0.0045, and an AUC value of 95.18%. These outcomes were achieved by utilizing just 12 out of the total 43 network features using the FS technique. Our experimental research, as presented in Tables 11a and 11b, indicates that the RF algorithm obtained accuracy rates of 99.30% and 99.14% in binary classification tasks using the complete feature set and a reduced feature set, respectively. With just 12 features and the appropriate hyperparameters taken from Table 7, the XGBoost model produced an FPR of 0.0048 in the UNSW-NB15 dataset's binary attack detection method, demonstrating a test accuracy of 99.04%.

Table 12a. UNSW-NB15 multiclass classification study utilizing all features.

ML Method	XGBoost	RF	SVM	LR	KNN	Ensemble
Test AC	98.26	98.25	97.90	97.32	97.61	98.57
Recall	98.26	98.25	97.90	97.32	97.61	98.57
Precision	98.21	98.09	97.50	96.46	97.61	98.46
F1	98.22	98.14	97.65	96.78	97.58	98.49
FPR	0.009	0.012	0.016	0.022	0.012	0.012
MCC	81.28	81.01	76.68	68.73	73.88	84.49
K	81.27	80.98	76.56	68.18	73.83	84.46
Training time	1820.49	188.84	36724.42	65059.11	0.41	213066.41
Prediction Time	30.50	6.19	4390.00	0.25	8000.13	7090.82
Total time	1850.99	195.03	41114.42	65059.36	8000.54	220157.23

Table 12b. UNSW-NB15 multiclass classification study utilizing reduced features

ML Method	XGBoost	RF	SVM	LR	KNN	Ensemble
Test AC	97.9	98.06	97.25	96.24	97.78	0.9811
Recall	97.9	98.06	97.25	96.24	97.78	0.9811
Precision	97.62	97.73	96.73	95.89	97.69	0.9795
F1	97.72	97.84	96.83	95.88	97.72	0.9797
FPR	0.014	0.016	0.023	0.02	0.014	0.017
MCC	76.98	78.63	68.85	59.36	76.01	0.7919
K	76.92	78.55	68.63	59.31	75.99	0.791
Training time	1021.42	100.75	33014.65	7520.58	14.18	546169.41
Prediction Time	24.56	5.77	4938.39	0.08	250.86	5388.81
Total time	1045.98	106.52	37953.04	7520.66	265.04	551558.22

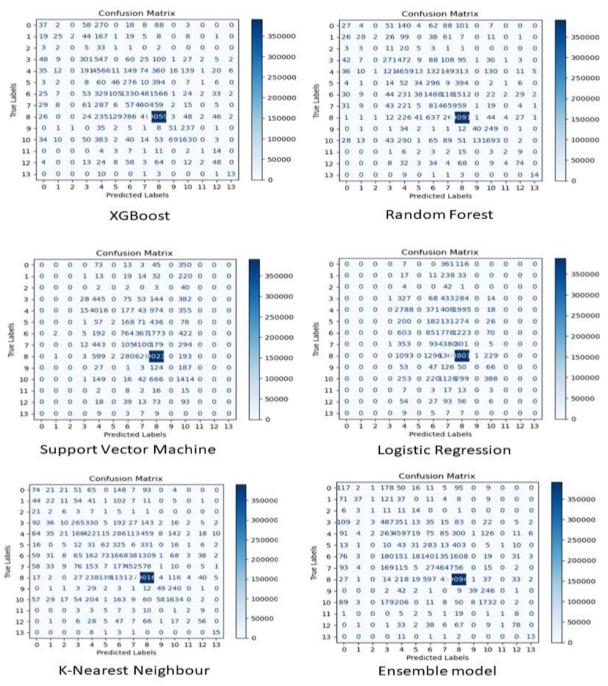

Figure 6. Confusion matrix of UNSW-NB15 multiclass classification (12 features)

Tables 12a and 12b present the average values of accuracy, FPR, MCC, precision, kappa statistics, recall, training duration, and F-score by utilizing different ML techniques on the multiclass classification scheme of the complete UNSW-NB15 dataset. Based on the experimental results shown in Table 12b and Figure 6, we have concluded that our ensemble technique outperforms previous methods, with a test accuracy of 98.11% and an FPR of 0.017. RF also performed well and achieved a test accuracy of 98.06% and an FPR of 0.0163 using a reduced feature set. ML-based multiclassification attack schemes typically require more time for training and prediction than binary-classification attack detection. However, as shown in Table 12b, our results demonstrate that applying a reduced feature set on the UNSW-NB15 dataset led to a significant reduction in training time for both LR and SVM. Consequently, as seen in Table 12b for the UNSW-NB15 dataset's category attack detection, the smaller feature set led to improvements in prediction time and training time as well as high detection accuracy.

To evaluate the efficacy of the proposed methodology, we utilized two extensive datasets of IoT IDSs, specifically UNSW-NB15, and IoTID20 in our experimental analysis. The UNSW_NB15 dataset has a higher size when compared to the IoTID20 dataset. The findings suggest that our methodology is adequately effective for implementing full and reduced feature sets for binary and multiclass attack detection techniques across all datasets under investigation.

3.4 Comparisons of Our Model with The Other Models

The main focus of our study is to examine the complete datasets of IDSs using FS methodology. However, it is equally important to assess the effectiveness of our approach by comparing it with other FS methodologies. Table 13 presents a comparison of the performance of our proposed models with other existing models for the binary classification of the UNSW-NB15 dataset. The UNSW-NB15 dataset in its entirety is used for the evaluation. The methodology that was proposed showed a test data accuracy of 99.13% by using only 12 features, outperforming other existing approaches as depicted in Table 13. A comparative analysis was conducted by referencing the research of (16)and (7), both of whom utilized the full UNSW-NB15 dataset in their various experimental studies. Furthermore, we compared our proposed model with the model introduced by (16) who experimented with several FS approaches offered by (13), (30), (31), (32), and (9). Our proposed strategy performs better than previously reported approaches with fewer features, as shown in Table 13.

Table 13. Our detection model performance compares with the previous model on the UNSW-NB 15 dataset.

Reference	**Model**	**No. of features**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**	**FPR**	**MCC (%)**	**K (%)**	**AUC (%)**
(7)	DL using ANN and SGDM	33	98.99	96.14	95.84	95.99	0.56	-	-	99.92
(16)	J48 and NB	All	96.63	-	-	-	1.04	-	-	-
(16)	RF	11 of (13)	95.16	-	-	-	-	-	-	-
(16)	RT (Random Tree)	5 of (30)	96.81	-	-	-	-	-	-	-

(16)	REPTree	8 of (31)	96.20	-	-	-	-	-	-	-
(16)	J48	17 of (32)	96.82	-	-	-	-	-	-	-
(16)	REPTree	18 of (9)	96.97	-	-	-	-	-	-	-
Proposed	Ensemble	12	99.13	99.13	99.13	99.13	0.0045	90.52	90.52	95.18
Proposed	RF	12	99.14	99.14	99.14	99.14	0.0041	90.62	90.62	95.00

Furthermore, we also conducted a comparative analysis between our proposed approach and various existing approaches that utilized the IoTID20 dataset for their experimental purposes. This comparison aimed to assess the efficacy of our approach in both binary and multiclass attack detection schemes. In comparison to the existing approaches, the suggested strategy obtained the best classification accuracy of 99.94% while utilizing fewer network features in the IoTID20 dataset. The comparison of our analysis utilizing the IoTID20 dataset with other existing methods for binary, category, and subcategory attack detection are presented in Tables 14, 15, and 16 respectively. The comparative analysis of the IoTID20 dataset, between the suggested methodology and the methodologies examined in existing literature, reveals that XGBoost outperforms other techniques in binary attack detection schemes. On the other hand, the stacking-based ensemble approach demonstrates much better performance in subcategory intrusion identification of the IoTID20 dataset.

Table 14. Our detection model performance comparison with a binary classification of the IoTID20 dataset.

Reference	**Model**	**No. of features**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**	**AUC (%)**	**FPR**	**MCC (%)**	**K (%)**
(20)	SLFN	All	98.44	-	-	-	-	-	-	-
(17)	KNN	28	99.66	99.70	99.70	99.70	99.00	0.027	-	-
(17)	Ensemble	28	99.98	99.90	99.90	99.90	99.90	0.008	-	-
(18)	Autoencoder	All	95.2	-	-	97.40	91.80	0.321	61.80	-
(33)	XGBoost	8	99.79	-	1	1	-	-	-	-
(33)	SVM	8	98.76	-	98.00	98.00	-	-	-	-
(3)	DCNN	All	99.91	99.87	99.38	99.62				
Proposed approach	XGBoost	16	99.94	99.94	99.94	99.94	99.71	0.0057	99.63	99.63
Proposed approach	Stacking based ensemble	16	99.88	99.88	99.88	99.88	99.46	0.0105	99.29	99.29

Table 15. Our detection model performance comparison with category classification of the IoTID20 dataset.

Reference	**Model**	**No. of features**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**	**FPR**	**MCC (%)**	**K (%)**

(3)	DCNN	All	98.38	97.73	97.83	97.77	-	-	-
(17)	Ensemble	28	99.70	-	-	-	-	-	-
Proposed approach	XGBoost	16	99.64	99.64	99.64	99.64	0.004	99.38	99.38
Proposed approach	Ensemble	16	98.65	98.64	98.65	98.64	0.012	97.68	97.68

Table 16. Our detection model performance comparison with subcategory classification of the IoTID20 dataset.

Reference	**Model**	**No. of features**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**	**FPR**	**MCC (%)**	**K (%)**
(24)	DT	All	88	88	88	88	-	-	-
(24)	Ensemble	All	87	87	87	87	-	-	-
(3)	DCNN	All	98.38	97.73	97.83	97.77	-	-	-
Proposed	Ensemble	16	92.66	92.53	92.66	92.51	0.01	90.9	90.89
Proposed	XGBoost	16	86.07	84.59	86.07	85.1	0.018	82.46	82.43

The performance outcomes of our proposed approach were compared with the previously existing approaches, as shown in Tables 13, 14, 15, and 16 for the two different datasets that we used in our experimental analysis. When compared to the previously used approaches, the comparison shows that our methodology produced better performance results. Furthermore, the current study's methodology utilizes a variety of evaluation criteria, as depicted in Tables 8a, 8b, 9a, 9b 10a, 10b, 11a, 11b, 12a, and 12b, to assess the prediction models generated by ML algorithms. This stands in contrast to earlier research that did not fully investigate all pertinent metrics. Hence, it can be inferred that the proposed methodology possesses the capability to effectively identify vulnerabilities in IoTIDSs.

CONCLUSION

This research study presents the application of the FS approach on the full version of IDS datasets to enhance IDSs on resource-constrained IoT devices with different ML techniques like XGBoost, RF, KNN, LR, SVM, and stacking-based ensemble approaches. Two current datasets were used to assess the models: IoTID20, which is intended to capture network traffic connected to IoT devices, and UNSW-NB15, which contains data on network traffic. In the present work, a combination of feature engineering techniques and dimensionality reduction methods have been used to enhance the performance. The outcome of the base classifiers must be temporarily stored by ensemble models to be used as the meta-classifier's input. The study introduces an ensemble learning-based model as a means to address the limitations apparent in individual ML classifiers. Two distinct dataset sizes have been used to assess our suggested methodology. The suggested approach in both scenarios enhances the detection rate. So, the findings of our research provide a base for creating strong and customized IDS that are perfect for managing the difficulties caused by IoT.

Our comprehensive investigation to enhance the IDSs for the IoT makes a valuable contribution to understanding IDSs. Our study provides essential findings and suggestions building a strong foundation for future investigations in IoT security. The devices connected to the IoT network are imitated resources and network devices are used extensively. So, the suggested methodology solving these problems and improving the IDS performance. This paper presents the use of a stacking-based ensemble method that is useful for network IDS datasets, especially when there is a significant class imbalance. By matching the experimental outcomes with earlier studies, it is found that the proposed strategy can enhance the efficiency of the IDS. In summary, this research makes a substantial contribution to the domain of ID in the IoT through the application of ML methodologies.

To extend future research endeavors, it is recommended to expand the implementation strategy to encompass experiments conducted on more complex datasets that incorporate DL techniques. While the suggested approach shows higher overall performance, further improvements are needed to identify specific attacks like Mirai and Scan in the IoTID20 dataset, Fuzzers, and DoS attacks in the UNSW-NB15 dataset configured in an IoT network utilizing the Contiki-NG operating system. Our future objectives entail the acquisition of data from a simulated environment of a Software-Defined Networking (SDN) testbed, followed by the application of DL techniques to conduct a comprehensive analysis and the features being selected using a statistical approach.

Supplementary Materials: Table S1: Outline of the ML-based experimentation on UNSW-NB15; Table S2: Outline of the ML-based experimentation on IoTID20

Author Contributions: Conceptualization, PB.; methodology, PS; formal analysis, PS.; investigation, PB.; data curation, PS.; writing original draft preparation, PB.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The reference for the dataset used in research is available within the article.

Conflicts of Interest: The authors declare no conflict of interest.

REFERENCES

[1] Ahanger TA, Aljumah A, Atiquzzaman M. State-of-the-art survey of artificial intelligent techniques for IoT security. Computer Networks. 2022:108771.

[2] Gyamfi E, Jurcut A. Intrusion detection in internet of things systems: a review on design approaches leveraging multi-access edge computing, machine learning, and datasets. Sensors. 2022;22(10):3744.

[3] Ullah S, Ahmad J, Khan MA, Alkhammash EH, Hadjouni M, Ghadi YY, et al. A new intrusion detection system for the internet of things via deep convolutional neural network and feature engineering. Sensors. 2022;22(10):3607.

[4] Moustafa N, Slay J, editors. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). 2015 military communications and information systems conference (MilCIS); 2015: IEEE.

[5] Kumar S, Gupta S, Arora S. Research trends in network-based intrusion detection systems: A review. IEEE Access. 2021;9:157761-79.

[6] Tsimenidis S, Lagkas T, Rantos K. Deep learning in IoT intrusion detection. Journal of network and systems management. 2022;30:1-40.

[7] Al-Zewairi M, Almajali S, Awajan A, editors. Experimental evaluation of a multi-layer feed-forward artificial neural network classifier for network intrusion detection system. 2017 International Conference on New Trends in Computing Sciences (ICTCS); 2017: IEEE.

[8] Agarwal A, Sharma P, Alshehri M, Mohamed AA, Alfarraj O. Classification model for accuracy and intrusion detection using machine learning approach. PeerJ Computer Science. 2021;7:e437.

[9] Kasongo SM, Sun Y. Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset. Journal of Big Data. 2020;7:1-20.

[10] Moustafa N, Slay J. The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. Information Security Journal: A Global Perspective. 2016;25(1-3):18-31.

[11] Thockchom N, Singh MM, Nandi U. A novel ensemble learning-based model for network intrusion detection. Complex & Intelligent Systems. 2023:1-22.

[12] Rajagopal S, Kundapur PP, Hareesha KS. A stacking ensemble for network intrusion detection using heterogeneous datasets. Security and Communication Networks. 2020;2020:1-9.

[13] Moustafa N, Slay J. A hybrid feature selection for network intrusion detection systems: Central points. arXiv preprint arXiv:170705505. 2017.

[14] Thakkar A, Lohiya R. Fusion of statistical importance for feature selection in Deep Neural Network-based Intrusion Detection System. Information Fusion. 2023;90:353-63.

[15] Vinayakumar R, Alazab M, Soman K, Poornachandran P, Al-Nemrat A, Venkatraman S. Deep learning approach for intelligent intrusion detection system. Ieee Access. 2019;7:41525-50.

[16] hew YJ, Lee N, Ooi SY, Wong K-S, Pang YH. Benchmarking full version of GureKDDCup, UNSW-NB15, and CIDDS-001 NIDS datasets using rolling-origin resampling. Information Security Journal: A Global Perspective. 2022;31(5):544-65.

[17] Albulayhi K, Abu Al-Haija Q, Alsuhibany SA, Jillepalli AA, Ashrafuzzaman M, Sheldon FT. IoT intrusion detection using machine learning with a novel high performing feature selection method. Applied Sciences. 2022;12(10):5015.

[18] Song Y, Hyun S, Cheong Y-G. Analysis of autoencoders for network intrusion detection. Sensors. 2021;21(13):4294.

[19] Islam N, Farhin F, Sultana I, Kaiser MS, Rahman MS, Mahmud M, et al. Towards Machine Learning Based Intrusion Detection in IoT Networks. Computers, Materials & Continua. 2021;69(2).

[20] Qaddoura R, Al-Zoubi AM, Almomani I, Faris H. A multi-stage classification approach for iot intrusion detection based on clustering with oversampling. Applied Sciences. 2021;11(7):3022.

[21] Prusa J, Khoshgoftaar TM, Seliya N, editors. The effect of dataset size on training tweet sentiment classifiers. 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA); 2015: IEEE.

[22] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine learning in Python. the Journal of machine Learning research. 2011;12:2825-30.

[23] Kang H, Ahn DH, Lee GM, Yoo J, Park KH, Kim HK. IoT network intrusion dataset. IEEE Dataport. 2019;10:q70p-q449.

[24] Ullah I, Mahmoud QH, editors. A scheme for generating a dataset for anomalous activity detection in iot networks. Advances in Artificial Intelligence: 33rd Canadian Conference on Artificial Intelligence, Canadian AI 2020, Ottawa, ON, Canada, May 13–15, 2020, Proceedings 33; 2020: Springer.

[25] Dahouda MK, Joe I. A deep-learned embedding technique for categorical features encoding. IEEE Access. 2021;9:114381-91.

[26] Chen R-C, Dewi C, Huang S-W, Caraka RE. Selecting critical features for data classification based on machine learning methods. Journal of Big Data. 2020;7(1):52.

[27] Saheed YK, Abiodun AI, Misra S, Holone MK, Colomo-Palacios R. A machine learning-based intrusion detection for detecting internet of things network attacks. Alexandria Engineering Journal. 2022;61(12):9395-409.

[28] Dhaliwal SS, Nahid A-A, Abbas R. Effective intrusion detection system using XGBoost. Information. 2018;9(7):149.
[29] Chen T, Guestrin C, editors. Xgboost: A scalable tree boosting system. Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining; 2016.
[30] Janarthanan T, Zargari S, editors. Feature selection in UNSW-NB15 and KDDCUP'99 datasets. 2017 IEEE 26th international symposium on industrial electronics (ISIE); 2017: IEEE.
[31] Moustafa N, Creech G, Slay J, editors. Anomaly detection system using beta mixture models and outlier detection. Progress in Computing, Analytics and Networking: Proceedings of ICCAN 2017; 2018: Springer.
[32] Anwer HM, Farouk M, Abdel-Hamid A, editors. A framework for efficient network anomaly intrusion detection with features selection. 2018 9th International Conference on Information and Communication Systems (ICICS); 2018: IEEE.
[33] Krishnan S, Neyaz A, Liu Q. IoT network attack detection using supervised machine learning. 2021.